Manuscript submitted to
AIMS' Journals
Volume **X**, Number **0X**, XX **200X**

doi:10.3934/xx.xx.xx.xx

pp. **X–XX**

SUBSPACE CLUSTERING BY (k, k)-SPARSE MATRIX FACTORIZATION

HAIXIA LIU*, JIAN-FENG CAI AND YANG WANG

Department of Mathematics,
The Hong Kong University of Science and Technology,
Clear Water Bay, Kowloon, Hong Kong, China.

ABSTRACT. High-dimensional data often lie in low-dimensional subspaces instead of the whole space. Subspace clustering is a problem to analyze data that are from multiple low-dimensional subspaces and cluster them into the corresponding subspaces. In this work, we propose a (k, k)-sparse matrix factorization method for subspace clustering. In this method, data itself is considered as the "dictionary", and each data point is represented as a linear combination of the basis of its cluster in the dictionary. Thus, the coefficient matrix is low-rank and sparse. With an appropriate permutation, it is also blockwise with each block corresponding to a cluster. With an assumption that each block is no more than k-by-k in matrix recovery, we seek a low-rank and (k, k)-sparse coefficient matrix, which will be used for the construction of affinity matrix in spectral clustering. The advantage of our proposed method is that we recover a coefficient matrix with (k, k)-sparse and low-rank simultaneously, which is better fit for subspace clustering. Numerical results illustrate the effectiveness that it is better than SSC and LRR in real-world classification problems such as face clustering and motion segmentation.

1. **Introduction.** There are huge amount of data generated and collected in our daily life. Despite of their different sources and natures, those seemingly structureless data often possess good structures. One commonly used structure is that high-dimensional data belonging to the same category lie in a low-dimensional subspace instead of arbitrarily distributed in the whole space. For example, hand-writings of the same digit with different rotations, translations or thickness are approximately in a low-dimensional subspace. Some other examples are human face image data, where the face images of each human under different illuminations form a low-dimensional subspace, and a video sequence with multiple moving objects, where each moving object in different frames belongs to a low-dimensional subspace. When multiple categories are presented, we will have a data set that is a union of low-dimensional subspaces, one category corresponding to a subspace. The problem of *subspace clustering* refers to extraction of the low-dimensional subspaces from such a data set with multiple categories. Many applications can be formulated as a subspace clustering problem, including face clustering and motion segmentation in computer vision, community clustering in social networks, and so on.

There are many available subspace clustering algorithms, including generalized principal component analysis (GPCA) [26], spectral curvature clustering [5], sparse subspace clustering (SSC) [8], low-rank representation (LRR) [14], low-rank subspace clustering (LRSC) [9], and low-rank and sparse subspace clustering (LRSSC) [27]. For more details

2010 *Mathematics Subject Classification.* Primary: 68T04; Secondary: 65F04.

Key words and phrases. Subspace clustering, Matrix factorization, (k, k)-sparse, Low-rank, k-support norm.

Y. Wang is supported in part by HKRGC grant 16306415.

J.-F. Cai is supported in part by HKRGC grant 16300616.

* Corresponding author.

about the algorithms of subspace clustering, interested readers may consult the survey paper [25]. Subspace clustering algorithms can be classified into four categories, namely, algebraic methods [6, 26, 27], iterative methods [1, 15], statistical methods [24, 23, 7, 21], and spectral clustering-based methods [28, 30, 11, 5, 8, 9, 30].

In spectral clustering-based methods, an affinity matrix W is constructed with the (i,j)-th entry measuring the similarity between the i-th and j-th data points. One of the challenges in applying spectral clustering-based methods is how to choose the affinity matrix W. Usually, W is constructed based on the coefficient matrix when the data set is represented as a linear combination of itself. Due to the structure of union of subspaces, the coefficient matrix possesses properties of low rank and sparsity, which are exploited in some state-of-the-art subspace clustering methods. In sparse subspace clustering (SSC) [8], a sparse coefficient matrix is assumed and sought. The low-rank representation (LRR) [14] try to find a low-rank representation of the coefficient matrix. Although SSC and LRR are both successful in subspace clustering, they are problematic under some circumstances, due to the reason that only one of the sparsity and the low-rank properties is used and the other is ignored. LRR has never been shown that it is successful without of an restrictive assumption of "independent subspace" [27]. SSC obtains an over-sparse coefficient matrix, based on which the affinity graph may not be a connected body for the data points from the same cluster [27, 16]. Moreover, the numerical experiments of Hopkins155 data show the instances where SSC fails are often different from that of LRR [27].

To fix these problems, one may consider both the low-rank and sparsity properties of the coefficient matrix. For this purpose, it is proposed in [27] to minimize a weighted sum of the nuclear norm and the ℓ_1-norm of the coefficient matrix. Since the nuclear norm promotes low rank and the ℓ_1-norm promotes the sparsity, it is expected that the coefficient matrix is simultaneously low-rank and sparse. However, it was shown in [18] that, when a matrix is jointly low-rank and sparse, minimizing a weighted sum of the nuclear norm and the ℓ_1-norm has little improvement over minimizing just one of the two norms.

We have to seek new regularization terms that can promote the simultaneously low-rank and sparse structure. In particular, the coefficient matrix in subspace clustering is jointly block sparse and low-rank, which is a linear combination of a few block sparse and rank-1 matrices. Based on this observation, the atomic norm [4] minimization is proposed in [22] to find the coefficient matrix. Unfortunately, the atomic norm optimization for jointly block sparsity and low rank recovery is NP-hard [22]. In this paper, we propose an algorithm to approximate the solution of the atomic norm minimization. Our algorithm is based on the low-rank factorization of the coefficient matrix, where the factors are sparse. By this way, we get a jointly sparse and low-rank coefficient matrix, which leads to effective subspace clustering. Numerical experiments demonstrate that our algorithm outperform state-of-the-art subspace clustering algorithms.

The rest of this paper is organized as follows. In Section 2, we present the formulation of the atomic norm for jointly block sparse and low-rank matrices. In Section 3, a new algorithm, called (k,k)-sparse matrix factorization, is proposed for matrix recovery of the corrupted data, which is applied for spectral-based subspace clustering in Section 4. Numerical results are given on two real-world clustering problems of face clustering and motion segmentation in Section 5. Finally, the paper is concluded in Section 6.

2. Jointly (k,k)-Sparse and Low-Rank Matrices and Subspace Clustering.

In this section, we present the atomic norm minimization for the construction of the coefficient matrix for subspace clustering.

2.1. **The Atomic Norm.** Our goal is to cluster a collection of data points which are drawn from a union of low-dimensional subspaces. Let

$$X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$$

be the data matrix, where each data point $\mathbf{x}_i \in \mathbb{R}^p, i = 1, \cdots, n$ is drawn from a union of t (t is unknown) linear subspaces $\{\mathcal{S}_i\}_{i=1}^t$ with the dimension of each subspace $d_i = \dim(S_i), i = 1, \cdots, t$, unknown. We assume that the dimension of each subspace is low enough such that it is smaller than the number of data points in this subspace. Then, each data points can be represented as a linear combination of data points from its cluster. Therefore, when each data point is represented as a linear combination of other data points in the same set, the coefficient matrix is block diagonal under an appropriate permutation, and each diagonal block corresponds to a subspace. Moreover, since each subspace is low dimensional, each diagonal block is also of low rank. More precisely, we use Z to denote the coefficient matrix of X under the linear representation of itself, i.e.,

$$X = XZ. \tag{1}$$

Then, in the ideal case, Z is a diagonal block low-rank matrix after suitable permutation. We assume that each nonzero block is no more than a k-by-k submatrix. In the following, we call the matrix (k,k)-sparse if the matrix is block-diagonal under an appropriate permutation and each nonzero block is no more than k-by-k submatrix.

To promote the (k,k)-sparse and low-rank properties simultaneously, we use the atomic norm [4, 22] minimization. Let

$$\mathcal{A}_{k,k} = \{\mathbf{u}\mathbf{v}^T : (\mathbf{u}, \mathbf{v}) \in \mathcal{S}_k^n \times \mathcal{S}_k^n\}$$

with

$$\mathcal{S}_k^n = \{\omega \in \mathbb{R}^n \mid \|\omega\|_0 \leq k, \|\omega\|_2 = 1\}.$$

In other words, $\mathcal{A}_{k,k}$ is a dictionary of infinite atorms, and each atom in $\mathcal{A}_{k,k}$ is a rank-1 matrix that contains at most k nonzero rows and k nonzero columns. The coefficient matrix Z can be represented as a linear combination of a small number of elements in $\mathcal{A}_{k,k}$

$$Z = \sum_{i=1}^{r} c_i A_i, \quad (c_i, A_i) \in \mathbb{R}_+ \times \mathcal{A}_{k,k}, \tag{2}$$

where r is the rank of Z. That is, Z has a sparse representation under dictionary $\mathcal{A}_{k,k}$. For a given arbitrary matrix M, its sparsity under dictionary $\mathcal{A}_{k,k}$ is defined

$$T_{k,k}^{(0)}(M) = \inf \left\{ p : M = \sum_{i=1}^{p} c_i A_i, \quad (c_i, A_i) \in \mathbb{R}_+ \times \mathcal{A}_{k,k} \right\}.$$

In subspace clustering, we want to find the coefficient matrix Z such that $T_{k,k}^{(0)}(Z)$ is as small as possible subject to suitable constraints. That is, we seek a coefficient matrix Z that uses as few as possible atoms in $\mathcal{A}_{k,k}$. However, the function $T_{k,k}^{(0)}$ is non-convex and discontinuous. Its best convex relaxation is the atomic norm [4, 22] as follows

$$T_{k,k}(M) = \inf \left\{ \sum_{i=1}^{p} c_i : M = \sum_{i=1}^{p} c_i A_i, \quad (c_i, A_i) \in \mathbb{R}_+ \times \mathcal{A}_{k,k} \right\}. \tag{3}$$

Instead of the minimization of $T_{k,k}^{(0)}(Z)$, a minimum $T_{k,k}(Z)$ is sought for the jointly (k,k)-sparse and low-rank coefficient matrix Z.

2.2. **Robust recovery to noises or outliers.** In applications, the data points are often corrupted by noise with possible outliers. To be precise, let $Y = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]$ be the data matrix, where each $\mathbf{y}_i \in \mathbb{R}^p, i = 1, \cdots, n$ is a corrupted data point with $\mathbf{y}_i = \mathbf{x}_i + \epsilon_i$, where $\mathbf{x}_i$ is the clear data point and ϵ_i is the corresponding error vector. Or, in matrix notations,

$$Y = X + \epsilon, \tag{4}$$

with $\epsilon = [\epsilon_1, \cdots, \epsilon_n]$. According to Equations (1) and (4), we have

$$Y = X + \epsilon = XZ + \epsilon = (Y - \epsilon)Z + \epsilon = YZ + (\epsilon - \epsilon Z) \triangleq YZ + E. \tag{5}$$

where $E = \epsilon - \epsilon Z$. We know the coefficient matrix $Z \in \mathbb{R}^{n\times n}$ is jointly low-rank and (k, k)-sparse. If we assume ϵ is sparse, then E is also sparse. This strategy was also used in [8, 28]. Therefore,

$$Y = YZ + E, \tag{6}$$

where Z is the coefficient matrix and E is the sparse error matrix. Hence, we can recover the jointly (k, k)-sparse and low-rank coefficient matrix via the following convex optimization

$$\begin{aligned} \min_{Z,E} \quad & T_{k,k}(Z) + \lambda_e \|E\|_1 \\ \text{s.t.} \quad & Y = YZ + E \end{aligned} \tag{7}$$

3. **Matrix recovery by (k, k)-Sparse Matrix Factorizaiton.** In this section, we present an algorithm based on matrix factorization to approximate a solution of (7).

3.1. **Matrix Factorization Approximation of the Atomic Norm.** In this section, we present an approximation of (7). Our starting point is the following theorem, which is a corollary of Theorem 11 in [22].

Theorem 3.1 (a Corollary of [22, Theorem 11])**.** *Let $Z \in \mathbb{R}^{n\times n}$ be a matrix. Then*

$$T_{k,k}(Z) = \inf_{Z=\sum_i \mathbf{u}_i\mathbf{v}_i^T} \frac{1}{2}\left(\sum_i (\|\mathbf{u}_i\|_k^{sp})^2 + \sum_i (\|\mathbf{v}_i\|_k^{sp})^2\right). \tag{8}$$

Here $\|\cdot\|_k^{sp}$ is the k-support norm [2] as follows:

$$\|\mathbf{v}\|_k^{sp} \triangleq \min\left\{\sum_{I\in\mathcal{G}_k} \|\mathbf{w}_I\|_2 : \operatorname{supp}(\mathbf{w}_I) \subseteq I, \sum_{I\in\mathcal{G}_k} \mathbf{w}_I = \mathbf{v}\right\},$$

where $\mathcal{G}_k$ denotes the set of all subsets of $\{1, 2, \cdots, n\}$ of cardinality at most k and $\operatorname{supp}(\mathbf{w}_I)$ is the support of the vector $\mathbf{w}_I$.

Proof. By [22, Theorem 11],

$$T_{k,k}(Z) = \inf\left\{\sum_i \|\mathbf{u}_i\|_k^{sp}\|\mathbf{v}_i\|_k^{sp} : Z = \sum_i \mathbf{u}_i\mathbf{v}_i^T\right\}. \tag{9}$$

Let the infimum on the right hand side of (8) be $S_1(Z)$, and that of (9) be $S_2(Z)$. According to the Cauchy-Schwarz inequality, we have

$$\begin{aligned} \sum_i \|\mathbf{u}_i\|_k^{sp}\|\mathbf{v}_i\|_k^{sp} &\le \sqrt{\sum_i (\|\mathbf{u}_i\|_k^{sp})^2}\sqrt{\sum_i (\|\mathbf{v}_i\|_k^{sp})^2} \\ &\le \frac{\sum_i (\|\mathbf{u}_i\|_k^{sp})^2 + \sum_i (\|\mathbf{v}_i\|_k^{sp})^2}{2} \end{aligned} \tag{10}$$

Therefore, $S_1(Z) \geq S_2(Z)$.

It remains to prove $S_1(Z) \leq S_2(Z)$. For a given arbitrary precision $\epsilon > 0$, let $\mathbf{v}_i$ and $\mathbf{u}_i$ be fixed such that $Z = \sum_i \mathbf{u}_i \mathbf{v}_i^T$ and $S_2(Z) + \epsilon \geq \sum_i \|\mathbf{u}_i\|_k^{sp} \|\mathbf{v}_i\|_k^{sp}$. We define, for any i,

$$\tilde{\mathbf{v}}_i = \mathbf{v}_i/\alpha_i,\ \tilde{\mathbf{u}}_i = \alpha_i \mathbf{u}_i, \quad \text{where} \quad \alpha_i = \sqrt{\|\mathbf{v}_i\|_k^{sp}/\|\mathbf{u}_i\|_k^{sp}}.$$

It can be easily checked that $Z = \sum_i \tilde{\mathbf{u}}_i \tilde{\mathbf{v}}_i^T$ and $\|\tilde{\mathbf{u}}_i\|_k^{sp} = \|\tilde{\mathbf{v}}_i\|_k^{sp}$, so

$$\begin{aligned} S_2(Z)+\epsilon &\geq \sum_i \|\mathbf{u}_i\|_k^{sp} \|\mathbf{v}_i\|_k^{sp} = \sum_i \|\tilde{\mathbf{u}}_i\|_k^{sp} \|\tilde{\mathbf{v}}_i\|_k^{sp} \\ &= \frac{\sum_i (\|\tilde{\mathbf{u}}_i\|_k^{sp})^2 + \sum_i (\|\tilde{\mathbf{v}}_i\|_k^{sp})^2}{2} \geq S_1(Z). \end{aligned}$$

Sending $\epsilon \to 0$ yields $S_2(Z) \geq S_1(Z)$. □

Note that the number of $\mathbf{u}_i$ and $\mathbf{v}_i$ are unknown in (8). If we fix this number to s, then we get an approximation of $T_{k,k}$ as follows

$$\min_{\substack{Z=UV^T \\ U,V \in \mathbb{R}^{n\times s}}} \frac{1}{2}\left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2\right),$$

with $\|U\|_{k,2} = \sqrt{\sum_{j=1}^s \left(\|\mathbf{u}_j\|_k^{sp}\right)^2}$ with $U = [\mathbf{u}_1, \cdots, \mathbf{u}_s]$ and $\|V\|_{k,2} = \sqrt{\sum_{j=1}^s \left(\|\mathbf{v}_j\|_k^{sp}\right)^2}$ with $V = [\mathbf{v}_1, \cdots, \mathbf{v}_s]$. Instead of (7), we solve its approximation

$$\begin{aligned} \underset{U,V,E}{\text{minimize}} \quad & \frac{1}{2}\left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2\right) + \mu_e \|E\|_1 \\ \text{s.t.} \quad & Y = YUV^T + E \end{aligned} \tag{11}$$

Model (11) is to find a (k,k)-sparse and low-rank representation by matrix factorization method. In the following, we call this method as (k,k)-*sparse matrix factorization*.

3.2. **Alternating Direction Method of Multipliers (ADMM).** In order to solve (11), we use the alternating direction method of multipliers (ADMM) [19]. By introducing an auxiliary variable, (11) is converted to an equivalence

$$\begin{aligned} \underset{Z,E,U,V}{\text{minimize}} \quad & \frac{1}{2}\left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2\right) + \mu_e \|E\|_1 \\ \text{s.t.} \quad & Y = YZ + E, \\ & Z = UV^T. \end{aligned} \tag{12}$$

Then the augmented Lagrangian function of (12) is

$$\begin{aligned} & L(Z,E,U,V,\lambda_1,\lambda_2) \\ = \ & \frac{1}{2}\left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2\right) + \mu_e \|E\|_1 \\ + \ & \langle \lambda_1, Y - YZ - E\rangle + \frac{\mu_1}{2}\|Y - (YZ+E)\|_F^2 \\ + \ & \langle \lambda_2, Z - UV^T\rangle + \frac{\mu_2}{2}\left\|Z - UV^T\right\|_F^2, \end{aligned} \tag{13}$$

where the Euclidean inner product of two matrices M and N is defined as $\langle M, N\rangle = \mathrm{tr}(MN^T)$, N^T is the transpose of the matrix N, and $\mathrm{tr}(\cdot)$ is the trace of a matrix.

Instead of the problem (12), ADMM solves

$$\max_{\lambda_1,\lambda_2} \min_{Z,E,U,V} L(Z,E,U,V,\lambda_1,\lambda_2)$$

by a gradient ascent algorithm, where the gradient is obtained approximately by one step of alternating direction minimization. The ADMM for (12) uses the following iteration

1. **Update Z:**

$$\begin{aligned} Z \quad \leftarrow \quad & \arg\min_{Z} \frac{\mu_1}{2} \left\| Y - (YZ + E) + \frac{\lambda_1}{\mu_1} \right\|_F^2 \\ & + \frac{\mu_2}{2} \left\| Z - UV^T + \frac{\lambda_2}{\mu_2} \right\|_F^2, \end{aligned} \tag{14}$$

2. **Update E and U, V, respectively:**

$$\begin{aligned} E \quad \leftarrow \quad & \arg\min_{E} \frac{\mu_1}{2} \left\| E - \left(Y - YZ + \frac{\lambda_1}{\mu_1} \right) \right\|_F^2 \\ & + \mu_e \|E\|_1 \\ = \quad & \operatorname{prox}_{\frac{\mu_e}{\mu_1}\|\cdot\|_1} \left(Y - YZ + \frac{\lambda_1}{\mu_1} \right), \end{aligned} \tag{15}$$

$$\begin{aligned} (U, V) \quad \leftarrow \quad & \arg\min_{U,V} \frac{\mu_2}{2} \left\| Z + \frac{\lambda_2}{\mu_2} - UV^T \right\|_F^2 \\ & + \frac{1}{2} \left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2 \right), \end{aligned} \tag{16}$$

3. **Update λ_1 and λ_2:**

$$\lambda_1 \leftarrow \lambda_1 + \mu_1 \left(Y - YZ - E \right), \tag{17}$$

$$\lambda_2 \leftarrow \lambda_2 + \mu_2 \left(Z - UV^T \right). \tag{18}$$

Subproblem (14) is a least squares problem and can be solved efficiently by linear system solvers. The solution of (15) is the entrywise soft-thresholding. However, it is not trivial to solve subproblem (16), and we employ an iterative method that will be discussed in Section 3.3 in detail. Algorithm 1 outlines the whole algorithm to solve (11) by ADMM.

Data: Initialize $E = 0, U = 0, V = 0, \lambda_1 = 0, \lambda_2 = 0$.
while *not convergence* **do**
    Update Z with Equation (14) ;
    Update E with Equation (15) and U, V with Equation (16), respectively;
    Update λ_1 with Equation (17) and λ_2 with Equation (18);
end

Algorithm 1: ADMM Algorithm to solve Model (11).

3.3. (k, k)**-Sparse Matrix Factorization.** In this subsection, we discuss efficient numerical solvers for solving subproblem (16) in ADMM. In particular, proximal alternating linearized minimization (PALM) [3] is applied.

For simplicity, we denote

$$\tilde{Z} = Z + \frac{\lambda_2}{\mu_2} \in \mathbb{R}^{n \times n}.$$

In (16), we find two matrices $U \in \mathbb{R}^{n\times s}$ and $V \in \mathbb{R}^{n\times s}$ that satisfy $\tilde{Z} \approx UV^T$ and UV^T is a (k,k)-sparse matrix, where s is a fixed integer approximating the rank. If we further define $E(U,V) = \frac{1}{2}\|\tilde{Z} - UV^T\|_F^2$, then (16) is rewritten as

$$[U,V] = \underset{\substack{U\in\mathbb{R}^{n\times s}\\ V\in\mathbb{R}^{n\times s}}}{\arg\min}\, E(U,V) + \frac{1}{2\mu}\left(\|U\|_{k,2}^2 + \|V\|_{k,2}^2\right) \tag{19}$$

Note that (19) is a non-convex optimization problem with respect to U and V. Finding the global minimum is generally challenging. Nevertheless, one standard approach for finding a local minimum of (19) is the Gauss-Seidel iteration, which is also known as alternating minimization or block coordinate descent. The alternating minimization for solving (19) is as follows:

$$\begin{cases} U \leftarrow \underset{U\in\mathbb{R}^{n\times s}}{\arg\min}\, E\left(U,V\right) + \frac{1}{2\mu}\|U\|_{k,2}^2 \\ V \leftarrow \underset{V\in\mathbb{R}^{n\times s}}{\arg\min}\, E\left(U,V\right) + \frac{1}{2\mu}\|V\|_{k,2}^2 \end{cases} \tag{20}$$

Unfortunately, the subproblems involved in (20) do not have close-form solutions, which means that nested iterative algorithms are needed. Furthermore, a necessary condition for the convergence of alternating minization is the minimum in each step is uniquely attained [29], and, otherwise, it is possible that the method may cycle without convergence [20].

If each subproblem in (20) is approximately solved by one step of proximal forward-backward iteration [19], then we get the proximal alternating linearized minimization (PALM) algorithm [3] as follows:

$$\begin{cases} U \leftarrow \mathrm{prox}_{\frac{1}{2\mu\zeta}\|\cdot\|_{k,2}^2}\left(U - \zeta(UV^T - \tilde{Z})V\right), \\ V \leftarrow \mathrm{prox}_{\frac{1}{2\mu\xi}\|\cdot\|_{k,2}^2}\left(V - \xi\left(VU^T - \tilde{Z}^T\right)U\right), \end{cases} \tag{21}$$

where $\mathrm{prox}_{\frac{1}{2\alpha}\|\cdot\|_{k,2}^2}$ is the proximity operators of $\frac{1}{2\alpha}\|\cdot\|_{k,2}^2$ as follows

$$\mathrm{prox}_{\frac{1}{2\alpha}\|\cdot\|_{k,2}^2}(H) = \underset{G\in\mathbb{R}^{n\times s}}{\arg\min}\left\{\frac{1}{2\alpha}\|G\|_{k,2}^2 + \frac{1}{2}\|G-H\|_F^2\right\}. \tag{22}$$

It is shown in [3] that (21) converges to a stationary point of (19), provided that the step sizes ζ and ξ satisfy certain constraints.

In order to get a practical algorithm, it remains to find an explicit expression of the proximity operator. To this end, let $G = [\mathbf{g}_1, \cdots, \mathbf{g}_s]$ and $H = [\mathbf{h}_1, \cdots, \mathbf{h}_s]$ in (22). Then we have

$$\begin{aligned} &\mathrm{prox}_{\frac{1}{2\alpha}\|\cdot\|_{k,2}^2}(H) \\ &\quad= \underset{G\in\mathbb{R}^{n\times s}}{\arg\min}\left\{\frac{1}{2\alpha}\|G\|_{k,2}^2 + \frac{1}{2}\|G-H\|_F^2\right\} \\ &\quad= \underset{\mathbf{g}_1,\cdots,\mathbf{g}_s}{\arg\min}\sum_{i=1}^{s}\left\{\frac{1}{2\alpha}(\|\mathbf{g}_i\|_k^{sp})^2 + \frac{1}{2}\|\mathbf{g}_i - \mathbf{h}_i\|_2^2\right\}. \end{aligned}$$

Since $\mathbf{g}_i$'s for $i = 1,\ldots,s$ are independent in the optimization in the last line of the above equation, the calculation of columns of $\mathrm{prox}_{\frac{1}{2\alpha}\|\cdot\|_{k,2}^2}(H)$ can be carried out independently. More precisely, if we write

$$\mathrm{prox}_{\frac{1}{2\alpha}\|\cdot\|_{k,2}^2}(H) = [\mathbf{p}_1,\ldots,\mathbf{p}_s],$$

then

$$\mathbf{p}_i = \arg\min_{\mathbf{g}_i} \left\{ \frac{1}{2\alpha}(\|\mathbf{g}_i\|_k^{sp})^2 + \frac{1}{2}\|\mathbf{g}_i - \mathbf{h}_i\|_2^2 \right\},$$

for $i = 1, \ldots, s$. A closed-form formula of $\mathbf{p}_i$ for a given $\mathbf{h}_i$ is provided by [2], which is outlined in Algorithm 2. For simplicity, we drop the index i in the vectors $\mathbf{h}_i$ and $\mathbf{p}_i$.

Data: $\mathbf{h} \in \mathbb{R}^n$ and the parameter α.
Result: $\mathbf{p} = \arg\min_{\mathbf{g}} \left\{ \frac{1}{2\alpha}(\|\mathbf{g}\|_k^{sp})^2 + \frac{1}{2}\|\mathbf{g} - \mathbf{h}\|_2^2 \right\}$

1 Let $\tilde{\mathbf{h}} = [\tilde{h}_1, \cdots, \tilde{h}_n]^T$, i.e., $\tilde{h}_i$ is the i-th largest element of $|\mathbf{h}|$. Let Π be the permutation matrix such that $\tilde{\mathbf{h}} = \Pi|\mathbf{h}|$. For simplicity, define $\tilde{h}_0 := +\infty$, $\tilde{h}_{n+1} := -\infty$ and $\gamma_{r,l} := \sum_{i=k-r}^{l} \tilde{h}_i$.

2 Find $r \in \{0, \cdots, k-1\}$ and $l \in \{k, \cdots, n\}$ such that

$$\frac{\tilde{h}_{k-r-1}}{\alpha+1} > \frac{\gamma_{r,l}}{l-k+(\alpha+1)(r+1)} \geq \frac{\tilde{h}_{k-r}}{\alpha+1},$$

$$\tilde{u}_l > \frac{\gamma_{r,l}}{l-k+(\alpha+1)(r+1)} \geq \tilde{g}_{l+1}.$$

3 Define

$$q_i = \begin{cases} \frac{\alpha}{\alpha+1}\tilde{h}_i & \text{if } i = 1, \cdots, k-r-1 \\ \tilde{h}_i - \frac{\gamma_{r,l}}{l-k+(\alpha+1)(r+1)} & \text{if } i = k-r, \cdots, l \\ 0 & \text{if } i = l+1, \cdots, n \end{cases}$$

4 Set $\mathbf{p} = [p_1, \ldots, p_n]^T$, where $p_i = \text{sign}(h_i)(\Pi^{-1}\mathbf{q})_i$.

Algorithm 2: The algorithm for proximity operator of $\frac{1}{2\alpha}(\|\cdot\|_{sp}^k)^2$ with input $\mathbf{h}$.

Since (19) is a non-convex optimization, the limit of our algorithm (21) depends on the initial guess . In general, the initialization with $U = 0$ and $V = 0$ is not good. Instead, we initialize U and V as follows. When (21) is called by the first iteration of Algorithm 1, we initialize U and V with

$$U = P_s \Sigma_s^{1/2}, \quad V = Q_s \Sigma_s^{1/2},$$

where (P_s, Q_s, Σ_s) are the singular value triplets correponding to the largest s singular value of $\tilde{Z}$. In the subsqeuent iterations of Algorithm 1, we use the U, V calculated in the previous iteration as the initial guess of current iteration and terminate the algorithm when

$$\max\left(\max\left(\left|U^{(l)}\left(V^{(l)}\right)^T - U^{(l-1)}\left(V^{(l-1)}\right)^T\right|\right)\right) < 2 \times 10^{-4}.$$

4. **Spectral clustering framework.** In this section, we present a framework of spectral subspace clustering based on the coefficient matrix $C = UV^T$ found by Algorithm 1 that solves (11).

As stated in the introduction, spectral clustering is very popular and is a very powerful tool for subspace clustering. In spectral clustering-based methods, an affinity matrix $W \in \mathbb{R}^{n \times n}$ is constructed with the (i, j)-th entry measuring the similarity between the i-th and j-th data points. First of all, we build a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$ with $\mathcal{V}$ being the set of nodes, $\mathcal{E}$ being the set of edges. W is the affinity matrix whose entries are the weights of an edge connecting two nodes. The weight equals 0 if there is no edge between two nodes. In an ideal graph, nodes from the same cluster are connected together and there are no edges

between the nodes from different clusters. One of the most challenges in applying spectral clustering-based methods is how to choose the affinity matrix.

We use the coefficient matrix $C = UV^T$ by Algorithm 1 to construct the affinity matrix. Recall that Algorithm 1 solves (11), where we use the data itself as the dictionary matrix, *i.e.* each sample is represented as a linear combination of the data points themselves. Moreover, we assume that each data point can be represented as a linear combination of the basis of those data points from the same cluster as itself. Ideally, for the coefficient matrix $C = UV^T$, the (i,j)-th entry $c_{i,j}$ of C is nonzero value if the i-th and j-th data points lie in the same cluster, otherwise, $c_{i,j} = 0$. Under an appropriate permutation, the matrix C is block-diagnal. We will construct the affinity matrix W based on the coefficient matrix C for spectral clustering. Note that it is possible that the coefficient matrix is not symmetric. Sometimes, the entries of the coefficient matrix are negative. In our experiments, the affinity matrix is chosen as $W = |C| + |C^T|$ with $w_{ij} = |c_{ij}| + |c_{ji}|$. Therefore, W is symmetric with nonnegative entries. Ideally, W is a block-diagonal matrix by an appropriate permutation with each block associated with one cluster.

Based on the affinity matrix W, we can apply normalized spectral clustering [17] for subspace clustering. More specifically, we compute the eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_t$ of the normalized Laplacian L_{sym} corresponding to the t (t is the number of clusters) smallest eigenvalues, and then we normalize the rows of $[\mathbf{x}_1, \cdots, \mathbf{x}_t]$ to norm 1 to get a matrix $\Phi = [\phi_1^T, \phi_2^T, \cdots, \phi_n^T]^T$; finally we cluster $\{\phi_j\}_{j=1}^n$ by K-means.

5. Numerical Results.

In this section, we evaluate our proposed method in face clustering and motion segmentation by experiments. Our method is also compared with the two state-of-the-art subspace clustering methods SSC [8] and LRR [14]. To have a fair comparison, we take out the post processing steps used in the original paper [8] of SSC and [14] of LRR, and we construct the affinity matrix by the coefficient matrix directly for subspace clustering.

5.1. Choice of s.

In our proposed method, matrix factorization is performed for matrix recovery. It is necessary to find an appropriate value of s. In practice, we do not have available knowledge about the reduced number of dimension s priorly, and a suitable s is always based on the given data. Generally speaking, s must be large enough to contain necessary information and small enough to take out the noises. In other words, if s is less than the real rank, then the data can not be adequately approximated, and if the reduced number s is equal or more than the real rank, then the reduction of the residual sum of square (RSS) is very slow. Therefore, in the plot of RSS versus different reduced numbers s, there is an inflection where s matches the proper rank number [12].

We use one experiment to investigate the relationship between the RSS and the number of the reduced dimension, and the RSS based on model (11) is defined as

$$\text{RSS}(U,V,E) = \|Y - YUV^T - E\|_F^2. \tag{23}$$

Figure 1 illustrates the plot of the residual sum of square (RSS) versus the reduced number s for $s = 5, \cdots, 10$ on one of the video sequences in Hopkins155 data. We see that there is an inflection point on the curve, which is chosen as s of our algorithm. We also observe that when s exceeds the inflection point, the RSS becomes stable. Therefore, our method is not sensive to the choice of the parameter s, as long as it is larger than the proper rank.

5.2. Face clustering.

Face clustering is to cluster the face images belonging to the same human subject into one cluster. In the following, we evaluate our proposed method on Extended Yale Database B [13]. This dataset contains face images for 38 human subjects

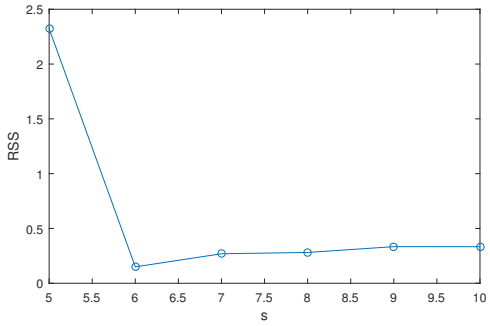

FIGURE 1. The plot of RSS vs. the reduced dimension s.

with 64 face images for each human subject under different illumination conditions, Figure 2 illustrates the face images of one human subject under different illumination conditions.

FIGURE 2. The ExtendedYale data B.

In our experiments, we test on the $t \in \{2, 3, 5, 8, 10\}$ subjects for subspace clustering. In order to save time and memory storage [10, 8], we resize each face image to 48×42 pixels from 192×168 pixels, and then further project the 2016-dimensional vector for each face image to $9 \times t$-dimensional subspace under singular value decomposition (SVD). For the parameters of SSC and LRR, we use the default values setting by the authors with $\alpha = 20$ for SSC and $\alpha = 0.18$ for LRR. For our proposed method, (k, k)-SMF, the parameters are setting as $\mu_1 = \alpha/\min_i \max_{j\neq i} \|\mathbf{y}_j\|_1$, $\mu_2 = \alpha$ and $\mu_e = \alpha/50000$ with $\alpha = 20$. Table 5.2 lists the mean (%) and median (%) of the error rate and the number of reduced dimension we assigned for $t \in \{2, 3, 5, 8, 10\}$ and $k \in \{3, 4\}$ in face clustering on Extended Yale dataset B. We see that in general our method achieves the minimum error among all the three methods.

5.3. **Motion segmentation.** In computer vision, motion segmentation is the process of separating regions, features, or trajectories from a video sequence into coherent subsets of space and time. These subsets correspond to independent moving objects in the scene. In the following, we use Hopkins155 dataset to evaluate our proposed method for motion segmentation. In this dataset, there are 156 video sequences including Checkerboard sequences, Traffic sequences and Articulated/non-rigid sequences. Each sequence is an independent data with the extracted feature points in all frames and there are 156 different datasets. In our numerical experiment, there is no pre/post processing on the dataset for all three methods. The parameters for SSC and LRR are also the default values setting by the authors. Since the outliers in the data have been manually removed, the overall error level

# Classes	mean/median	SSC	LRR	$(3,3)$-SMF		$(4,4)$-SMF	
				error	s	error	s
2	mean	15.83	6.37	**3.38**	18	3.53	18
	median	15.63	6.25	**2.34**		**2.34**	
3	mean	28.13	9.57	6.19	25	**6.06**	25
	median	28.65	8.85	**5.73**		**5.73**	
5	mean	37.90	14.86	11.06	35	**10.04**	35
	median	38.44	14.38	9.38		**9.06**	
8	mean	44.25	23.27	23.08	50	**22.51**	50
	median	44.82	**21.29**	27.54		26.06	
10	mean	50.78	29.38	25.36	65	**23.91**	65
	median	49.06	32.97	**27.19**		27.34	

TABLE 1. The error rate (mean % and median %) for face clustering on Extended Yale dataset B.

is low, we set the parameters of error $\mu_e = \alpha/10$ with $\alpha = 80$. Table 5.3 lists the error rate (mean %/median %) for motion segmentation on Hopkins155 dataset with the number of reduced dimension $s = 10$. From Table 5.3, we see that our method has a smaller mean error than both SSC and LRR.

	SSC	LRR	$(3,3)$-SMF	$(4,4)$-SMF
Mean	9.28	8.43	**6.61**	7.16
Median	**0.24**	1.54	1.20	1.32

TABLE 2. The error rate (mean %/median %) for motion segmentation on Hopkins155 dataset.

6. **Conclusion.** In this work, we have proposed a (k,k)-sparse matrix factorization method for subspace clustering. This method consists of a matrix recovery by (k,k)-sparse matrix factorization and a spectral clustering. In matrix recovery by (k,k)-sparse matrix factorization, we recover a low-rank and (k,k)-sparse coefficient matrix, which is used to construct the affinity matrix for spectral clustering. Our (k,k)-sparse matrix factorization algorithm is robust to outliers contained in the corrupted data. Our method is evaluated by experiments on the datasets of Extended Yale dataset B for face clustering and Hopkins155 dataset for motion segmentation. Numerical results demonstrate that our proposed method is better than the two state-of-the-art subspace clustering methods SSC and LRR.

There are several possible future research directions. One is to establish the convergence and theoretical guarantee of the proposed algorithm for subspace clustering. Another one is to investigate theoretical results of the atomic norm minimization for recovering low-rank and (k,k)-sparse matrix.

REFERENCES

[1] P. K. AGARWAL AND N. H. MUSTAFA, *k-means projective clustering*, in Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM, 2004, pp. 155–165.

[2] A. ARGYRIOU, R. FOYGEL, AND N. SREBRO, *Sparse prediction with the k-support norm*, in Advances in Neural Information Processing Systems, 2012, pp. 1457–1465.

[3] J. BOLTE, S. SABACH, AND M. TEBOULLE, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Mathematical Programming, 146 (2014), pp. 459–494.

[4] V. CHANDRASEKARAN, B. RECHT, P. A. PARRILO, AND A. S. WILLSKY, *The convex geometry of linear inverse problems*, Foundations of Computational mathematics, 12 (2012), pp. 805–849.
[5] G. CHEN AND G. LERMAN, *Spectral curvature clustering (scc)*, International Journal of Computer Vision, 81 (2009), pp. 317–330.
[6] J. P. COSTEIRA AND T. KANADE, *A multibody factorization method for independently moving objects*, International Journal of Computer Vision, 29 (1998), pp. 159–179.
[7] H. DERKSEN, Y. MA, W. HONG, AND J. WRIGHT, *Segmentation of multivariate mixed data via lossy coding and compression*, in Electronic Imaging 2007, International Society for Optics and Photonics, 2007, pp. 65080H–65080H.
[8] E. ELHAMIFAR AND R. VIDAL, *Sparse subspace clustering: Algorithm, theory, and applications*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), pp. 2765–2781.
[9] P. FAVARO, R. VIDAL, AND A. RAVICHANDRAN, *A closed form solution to robust subspace estimation and clustering*, in 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2011, pp. 1801–1807.
[10] J. FENG, Z. LIN, H. XU, AND S. YAN, *Robust subspace segmentation with block-diagonal prior*, in IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 3818–3825.
[11] A. GOH AND R. VIDAL, *Segmenting motions of different types by unsupervised manifold clustering*, in IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–6.
[12] L. N. HUTCHINS, S. M. MURPHY, P. SINGH, AND J. H. GRABER, *Position-dependent motif characterization using non-negative matrix factorization*, Bioinformatics, 24 (2008), pp. 2684–2690.
[13] K. LEE, J. HO, AND D. KRIEGMAN, *Acquiring linear subspaces for face recognition under variable lighting*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005), pp. 684–698.
[14] G. LIU, Z. LIN, AND Y. YU, *Robust subspace segmentation by low-rank representation*, in Proceedings of the 27th international conference on machine learning, 2010, pp. 663–670.
[15] L. LU AND R. VIDAL, *Combined central and subspace clustering for computer vision applications*, in Proceedings of the 23rd international conference on Machine learning, ACM, 2006, pp. 593–600.
[16] B. NASIHATKON AND R. HARTLEY, *Graph connectivity in sparse subspace clustering*, in Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 2137–2144.
[17] A. Y. NG, M. I. JORDAN, AND Y. WEISS, *On spectral clustering: Analysis and an algorithm*, Advances in neural information processing systems, 2 (2002), pp. 849–856.
[18] S. OYMAK, A. JALALI, M. FAZEL, Y. C. ELDAR, AND B. HASSIBI, *Simultaneously structured models with application to sparse and low-rank matrices*, Information Theory, IEEE Transactions on, 61 (2015), pp. 2886–2908.
[19] N. PARIKH AND S. BOYD, *Proximal algorithms*, Foundations and Trends in optimization, 1 (2013), pp. 123–231.
[20] M. J. D. POWELL, *On search directions for minimization algorithms*, Mathematical Programming, 4 (1973), pp. 193–201.
[21] S. R. RAO, R. TRON, R. VIDAL, AND Y. MA, *Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories*, in IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.
[22] E. RICHARD, G. R. OBOZINSKI, AND J.-P. VERT, *Tight convex relaxations for sparse matrix factorization*, in Advances in Neural Information Processing Systems, 2014, pp. 3284–3292.
[23] Y. SUGAYA AND K. KANATANI, *Geometric structure of degeneracy for multi-body motion segmentation*, in Statistical Methods in Video Processing, Springer, 2004, pp. 13–25.
[24] M. E. TIPPING AND C. M. BISHOP, *Mixtures of probabilistic principal component analyzers*, Neural computation, 11 (1999), pp. 443–482.
[25] R. VIDAL, *A tutorial on subspace clustering*, IEEE Signal Processing Magazine, 28 (2010), pp. 52–68.
[26] R. VIDAL, Y. MA, AND S. SASTRY, *Generalized principal component analysis (gpca)*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005), pp. 1945–1959.
[27] Y.-X. WANG, H. XU, AND C. LENG, *Provable subspace clustering: When LRR meets SSC*, in Advances in Neural Information Processing Systems, 2013, pp. 64–72.
[28] J. YAN AND M. POLLEFEYS, *A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate*, in Computer Vision–ECCV 2006, Springer, 2006, pp. 94–106.
[29] W. I. ZANGWILL, *Nonlinear programming: a unified approach*, vol. 196, Prentice-Hall Englewood Cliffs, NJ, 1969.
[30] T. ZHANG, A. SZLAM, Y. WANG, AND G. LERMAN, *Hybrid linear modeling via local best-fit flats*, International journal of computer vision, 100 (2012), pp. 217–240.

E-mail address: mahxliu@ust.hk
E-mail address: jfcai@ust.hk
E-mail address: yangwang@ust.hk